On-Board Feature Extraction from Acceleration Data for Activity Recognition

Atis Elsts, Ryan McConville, Xenofon Fafoutis, Niall Twomey,
Robert Piechocki, Raul Santos-Rodriguez and Ian Craddock
University of Bristol

Abstract

Modern wearable devices are equipped with increasingly powerful microcontrollers and therefore are increasingly capable of doing computationally heavy operations, such as feature extraction from sensor data. This paper quantifies the time and energy costs required for on-board computation of features on acceleration data, the reduction achieved in subsequent communication load compared with transmission of the raw data, and the impact on daily activity recognition in terms of classification accuracy. The results show that platforms based on modern 32-bit ARM Cortex-M microcontrollers significantly benefit from on-board extraction of time-domain features. On the other hand, efficiency gains from computation of frequency domain features at the moment largely remain out of their reach.

Categories and Subject Descriptors

I.5 [**Pattern Recognition**]: Models, Applications

General Terms

Algorithms, Performance

Keywords

Wearables, Embedded Systems

1 Introduction

Recognition of *activities of daily living* (ADL) using wearable accelerometer sensors is a core enabling technology for healthcare, fitness, and ambient assisted living applications [2]. ADL classification algorithms usually do not directly operate on the raw data, but rely on various *features* computed from that data. At the moment, acceleration data often is collected on wearable sensor nodes powered by batteries and equipped with low-power embedded microcontrollers, while the classification of ADL themselves typically happens on computationally more powerful devices, such as smartphones or computational elements in the cloud. However, recent generations of wearable devices are increasingly capable of doing the feature extraction and data preprocessing on their own; rather than fully transmitting the raw data, they are able to compute the features on-board and transmit just the results, which then can be used as input for activity classification on a more powerful device. This approach has a potential to save energy & increase battery life.

Contribution This paper presents a comparative performance evaluation study of a large number of features from acceleration data; the costs of their computation are compared on multiple low-power microcontroller platforms.

Methods We quantify the costs and benefits of a number of different features in terms of associated energy consumption and their importance for activity recognition. The features include both time domain and frequency domain features – for example, the mean, the median, and histograms of the raw acceleration data. For this study, we consider multiple recent low-power hardware platforms suitable for wearables: two ARM Cortex-M3 based and one ARM Cortex-M4F based. Additionally in the comparison we include Texas Instruments *msp430* MCU and ARM Cortex-A53, which serve as the lower and upper bounds of performance. For each of the platforms, we evaluate the energy cost of computing each feature and the cost of transmitting the data obtained as the result of that computation. For each of the features, we calculate its mutual information score on the ground-truth labels of activities, and its usefulness for activity recognition using a random forest classifier. Finally, we present system-level energy-consumption results on two platforms – SPW-2 [5], a wearable device with Texas Instruments CC2650 System-on-Chip (SoC) that is based on ARM Cortex-M3 core, and the *msp430*-based Zolertia Z1.

Results The intention of this paper is twofold: first, it is to serve designers of new wearable platforms, and the networked embedded system research community in general, by facilitating informed choices about microcontroller selection and system-level software design. Second, it is to provide cost/benefit analysis of on-board computation of various features known from the research literature. Some particular highlights of the results are:

- On-board computation of both time-domain and frequency-domain features on modern microcontrollers is feasible while keeping the duty cycle low. On-board computation of time-domain features in particular leads to extensive energy savings compared with transmission

of raw data.

- The historically accurate cost/performance tradeoff between 8/16-bit and 32-bit platforms [11] is now obsolete, as recent 32-bit Cortex-M based microcontrollers show unequivocally better results both in performance (by one or more orders of magnitude) and energy efficiency than older 16-bit systems.
- Neither the model of the MCU core, nor its system clock frequency determine its performance and energy consumption. In particular, for some features the differences in performance between two different generations of Cortex-M3 based microcontrollers dwarf the differences between two single-generation Cortex-M3 MCU and Cortex-M4F based microcontrollers.
- Using Cortex-M4F based MCU with floating-point co-processor may lead to significant performance-per-MHz *reduction* on integer-only features.

2 Related work

2.1 Cost/benefit analysis of features

An activity recognition system that is based resource-constrained sensing systems needs to balance the trade-off between the accuracy of the output knowledge and the cost of collecting the data. This trade-off is reported in the literature as the cost-accuracy conflict [10]. A number of works attempt to solve this conflict proposing the assignment of a cost value to each potential feature; the goal of the learning process is to jointly minimize both the cost and classification error. This cost value can be an abstract measure [9], or can depend on computational costs [19] or financial costs [22]. However, the analysis is often done on mobile phones [4], without taking into account the specifics of more energy-constrained embedded systems – the focus of this paper.

2.2 On-board computation on wearables

At the moment majority of wearables belong to one of two classes. One of those is the class of highly-efficient systems with low-power microcontrollers and battery lifetimes of weeks or more. For this class, 16-bit microcontrollers are still used both in commercial devices (*e.g.*, SHIMMER [12]) and for research – as a brief perusal of recent papers from SenSys and EWSN confirms [7, 6, 15]. The other extreme is computationally more capable systems, often with > 1 GHz MCU, but with higher energy consumption and more frequent requirements for recharge. This class contains most of commercial smartwatches and smartphones, as well as many research prototypes [8, 18]. The contribution of the present paper is to show that the more powerful modern 32-bit MCUs combine the best of both worlds: high efficiency with sufficient power for on-board processing. The 16-bit *vs.* 32-bit energy/power tradeoff [11] is now obsolete.

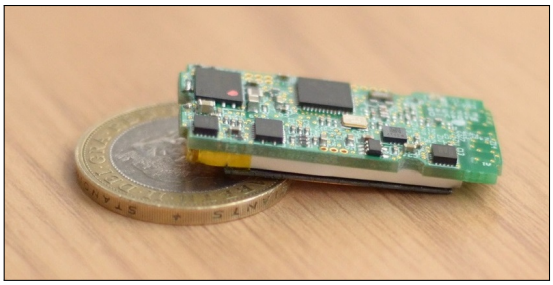


Figure 1: SPW-2: A CC2650 SoC based wearable accelerometer sensor [5].

3 Platforms

We primarily evaluate on-board feature extraction on 32-bit Cortex-M processors, commonly found in modern IoT platforms. SPW-2 [5], shown in Fig. 1, is a wrist-worn accelerometer-based wearable sensor that is designed for long-term residential monitoring with minimum maintenance. SPW-2 is equipped with the CC2650, a multi-standard 2.4 GHz ultra-low power wireless System-on-Chip that is based on the Cortex-M3 processor.

In addition to SPW-2, we perform the evaluation on a series of commercial IoT platforms, namely Zolertia Zoul, Nordic nRF52-DK, Zolertia Z1 and the Raspberry Pi 3B (summarized in Table 1). These platforms do not have the form-factor of a wearable sensor; however, with the exception of the Raspberry Pi, their MCUs can be potentially used in wearables. Zolertia Zoul is based on the CC2538, a System-on-Chip for 2.4-GHz IEEE 802.15.4 that is also equipped with the Cortex-M3 processor [25]. Performance comparisons between CC2538 and CC2650, *i.e.* two platforms with the same processing unit, would help to identify the importance of the other elements within the System-on-Chip. Nordic nRF52-DK is a development kit for the nRF52832 System-on-Chip [16]. This platform employs the Cortex-M4F processing unit; in contrast to the Cortex-M3 based platforms, it has an extended instruction set and a Floating-Point Unit (FPU).

The benchmarks also include results obtained on Zolertia Z1 and Raspberry Pi Model 3B. Zolertia Z1 [24] has

Table 1: Comparison of the hardware platforms and microcontrollers

Platform name	MCU / SoC	MCU core	MCU frequency	Nonvolatile memory	RAM	FPU present	Active-mode MCU current
Zolertia Z1 [24]	MSP430F2617	msp430	8 MHz	116 kB	8 kB	–	4.2 mA
Zolertia Zoul [25]	CC2538	Cortex-M3	32 MHz	512 kB	32 kB	–	13.0 mA
SPW-2 [5]	CC2650	Cortex-M3	48 MHz	128 kB	20 kB	–	2.9 mA
Nordic nRF52-DK [16]	nRF52832	Cortex-M4F	64 MHz	512 kB	64 kB	+	3.3 mA
Raspberry Pi 3B [17]	BCM2837	Cortex-A53	1200 MHz	Multiple GB[1]	1 GB[1]	+	221.0 mA[2,3]

[1] Separate from the CPU
[2] Assuming single active core @ 1.2 GHz
[3] At lower voltage than other platforms: ≈ 1.0 V

a 16-bit *msp430* processor, typical for previous generation low-power embedded platforms. In contrast, the Raspberry Pi Model 3B [17] serves as an upper performance bound. The Raspberry Pi employs the BCM2837 System-on-Chip which incorporates a Cortex-A53 processor. We node that the BCM2837 is too energy-hungry to be supported from the batteries of long-lifetime wearable sensing platforms.

4 The dataset and features

4.1 The dataset

This paper evaluates feature extraction on the SPHERE challenge dataset [20], a publicly available research dataset that includes ground-truth labels of activities. The dataset captures wrist-worn accelerometer data (20 Hz sampling rate, ± 4 g range) from 10 participants, and has already been used in a number of activity recognition works [1, 14]. In order to avoid having to deal with missing data, we use a subset of the dataset: the activities of six participants, each of which has $< 5\,\%$ of samples missing because of lost over-the-air packets, and quantize the readings as 8-bit integers.

4.2 The features

Activity recognition using wrist-worn accelerometer data is typically based on features calculated from the raw accelerometer data. We decided on a set of features (Table 2) to investigate, many of which are commonly used across existing studies of activity recognition [21, 23]. To capture the temporal nature of activities, the features are calculated over a window of time. The chosen window size in this paper is 64 samples (3.2 sec at 20 Hz sampling rate); this is a common window size for activity recognition. Each feature is computed once per second, *i.e.*, sequential rolling windows are largerly overlapping. The features are computed both in the time domain and frequency domain; for the latter, the Fourier transform is first applied on the window before calculating the feature.

We evaluate the usefulness of the features in two dimensions. The first dimension is their discriminating power for activity recognition; the second is the energy cost associated with the extraction and transmission of each feature. Thus a balance must be struck between the effectiveness of the feature and the energy required for its extraction.

4.3 Implementation of the features

We implement the features as a stand-alone C programming language library[4]. The code is fully portable and does not contain any ARM Cortex specific functionality. Using an ARM-specific library such as CMSIS[5] could potentially further improve the speed of the code; however, due to its non-portability and large binary code size it is not appropriate for some of the test platforms. The computations of trigonometric functions and the entropy are sped up using pre-computed lookup tables. For the evaluation, we compile the library with `msp430-gcc 4.7.2` for Zolertia Z1, `arm-linux-gnueabihf-gcc 4.9.2` for Raspberry Pi, and `arm-none-eabi-gcc 4.9.3` for the ARM Cortex-M based platforms. Optimization option `-O2` is selected.

[4] https://github.com/IRC-SPHERE/embedded-features

[5] http://www.keil.com/cmsis

5 Performance evaluation

5.1 Feature-specific cost evaluation

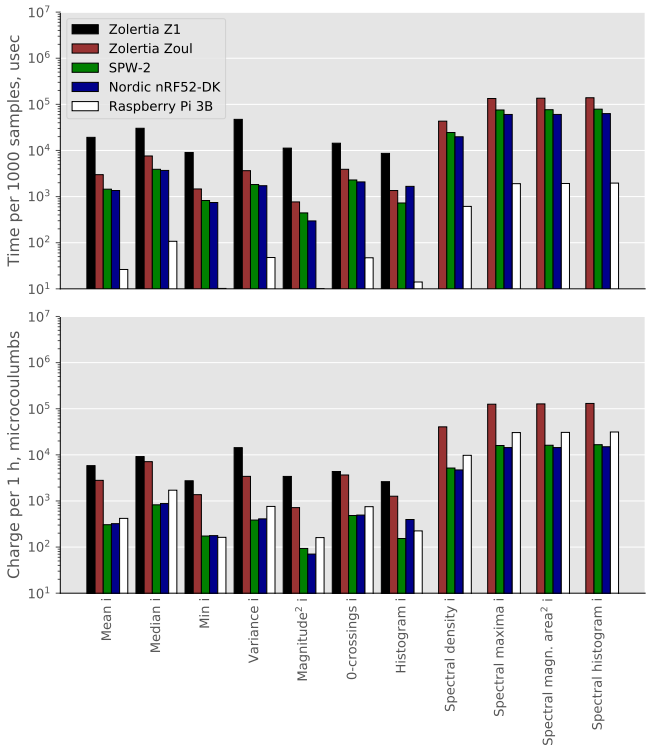
The first experiment is to evaluate the cost (in terms of time and electric charge) of computing each specific feature on the different microcontrollers. To this end, we design and run a test application on each of the platforms under consideration. The application goes through a list of features and computes each one on the input data (Section 4.1). For better accuracy, for features that are fast to compute (≤ 0.1 seconds required) the computation is repeated multiple times and the average computation time is used. The spectral features are not evaluated on the *msp430*-based system as too complex. The execution time is measured in software; in contrast, the electric charge is extrapolated from time measurements. We provide electric charge instead of energy, as the latter is heavily dependent on the operating voltage, so mainly characterizes the performance of the platform as whole, not the MCU.

The results are shown in Fig. 2. First of all, there is an order of magnitude difference between Cortex-M and *msp430*, both in speed and in energy consumption (using the charge as a proxy). There is also an order of magnitude difference in energy consumption between the older Cortex-M based platform and the two newer ones. When compared with Cortex-A, the two newer Cortex-M MCU differ by one or two magnitudes in speed, but have similar or smaller requirements for the *charge*. Still, the operating voltage of Cortex-A (≈ 1.0 V) is much lower than any of the other MCU (typically in the range from 1.7–2.0 V up to 3.6–3.8 V), the Cortex-A demonstrates the best *energy* consumption overall, by a small margin. This is not surprising, as the strength of the other microcontrollers lies in their highly efficient low power modes.

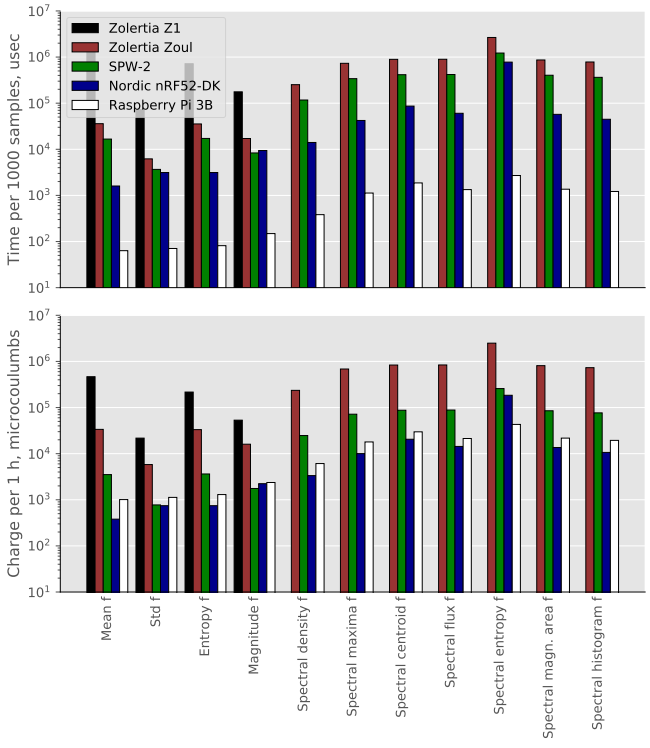
Figure 3 shows the comparison of the per-MHz performance of different groups of feature on the different microcontrollers. Several interesting observations can be made. First, the performance on the two Cortex-M3 based platforms is not identical, even though they both are made by the same manufacturer and meant for the same application area. Second, while the Cortex-M4F based microcontroller with FPU on the average computes floating point much faster, it is less performing for integer-only operations (up to 3 times slower per MHz), up to the point that it is slower even for some features that involve floating point, but also require a lot of integer operations (such as the standard deviation).

Subsequently, the amounts of the data produced by the features are evaluated in order to quantify the transmission cost. We assume the data is efficiently encoded: for integers, CBOR [3] is used, for floating point numbers: their size is reduced to 16 bits. 100 % radio overhead is assumed for radio transmissions: *i.e.*, the radio stays on for twice as long as required to transmit the payload data itself. This is a typical overhead in BLE and IEEE 802.15.4 6LoWPAN networks.

Figure 4a shows that the time-domain features require computational costs that are proportionally negligible compared with the transmission cost of the original data: $< 10\,\%$ for all of them, $< 1\,\%$ for most. On the other hand, the output data size of a single feature is up to an order of magnitude smaller than the original data size. Consequently, computing and sending out time-domain features instead of the raw data leads to large energy savings. In contrast, spec-



(a) Integer-based features



(b) Floating-point features

Figure 2: Time and electrical charge required to compute the features on the test platforms. The time is measured; the charge is calculated by multiplying the time with the MCU active-mode current given in Table 1.

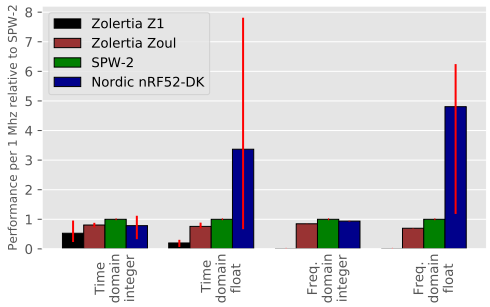


Figure 3: Performance of the different low-power platforms per 1 MHz relative to SPW-2. The error bars show the best and the worst performing feature of each group. Note that features with few FPU-specific operations (e.g. `magnitude f`) show *reduced* performance on the system with FPU.

tral features (Fig. 4b) all require computation of the Fourier transform, which is relatively heavy on its own, and often incur high computational costs on top of that. Some of these features, such as the spectral energy density and the spectral histogram, *increase* the output data size, therefore are wholly unsuitable for generation of data that is transmitted over the air. However, some of them, especially the features with the smaller computational costs, may be suitable for on-board *classification* of activities.

The results are generalizable to systems that store data locally. In particular, the energy costs for writing the data on the the flash storage unit on SPW-2 are on the same order of magnitude as for transmitting the data via the CC2650 radio.

5.2 Importance of the features

We use two common methods to measure the importance of features for activity recognition. The first method consists of calculating the Mutual Information (MI) [13] metric between each feature and the activity labels. The second involves training a number of Random Forest (RF) classifiers on the dataset, and measuring (1) the importance of each feature in classifiers allowed to use all features (Fig. 6) and (2) the classification accuracy of each feature taken separately (Table 2). For these tasks, we select (by random sampling) a balanced subset of the dataset (Section 4.1) with 13 min of each of 4 core activities: walking, standing, sitting and lying.

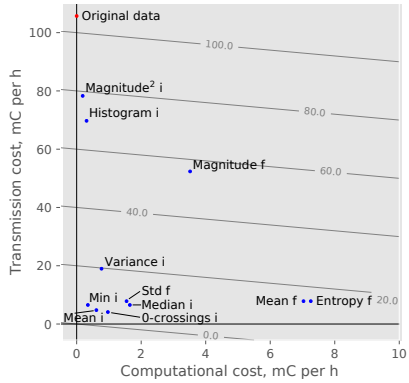
5.2.1 *Mutual information*

The MI of two random variables measures their mutual dependence; higher values imply higher dependency, zero implies statistical independence. In this work, the total MI of features consisting of multiple parts (*e.g.*, the histograms) is approximated as the sum of the MI of their parts.

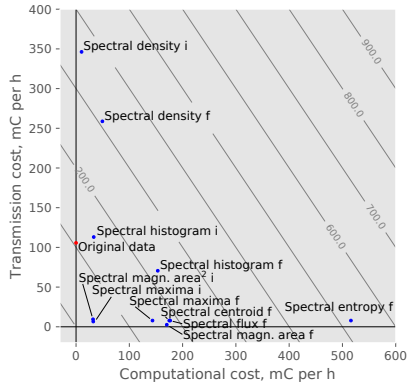
Figure 5 shows the calculated informativeness of the various features w.r.t. activity labels; features with a combined MI value less than 0.01 are excluded for clarity. Various statistical time domain features cluster together in terms of importance and energy cost. On the other hand, spectral histograms and the spectral density have significantly higher mutual information, but with an increased energy cost.

5.2.2 *Classification accuracy*

We look at how well the features perform at the activity recognition task, which is defined as: given the extracted features at a given timepoint t, and an associated activity label, build a model that given the extracted feature(s) for t deter-



(a) Time domain features



(b) Frequency domain features

Figure 4: Computation and transmission costs of the various features compared with the original data, based on measurements on the SPW-2 platform.

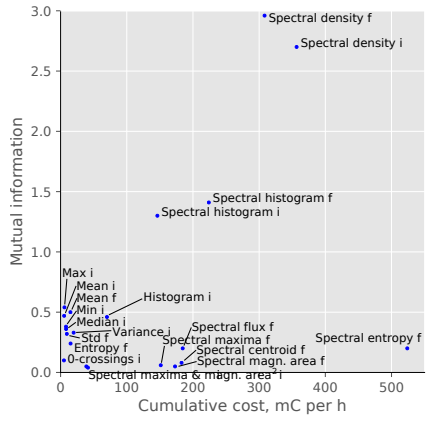


Figure 5: The Mutual Information (MI) score *vs* the energy cost.

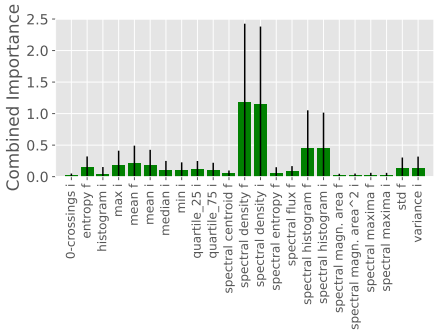


Figure 6: The importance (y-axis) of each feature (x-axis) as determined from the Random Forest (RF) classifier. Errorbars show standard deviation.

Table 2: RF classifier trained and evaluated on each individual feature.

Feature	Accuracy (=F1 score)	Extraction cost, mC/h	Total cost, mC/h
Max i	0.66	0.35	5.83
Spectral histogram f	0.63	153.38	223.96
Spectral histogram i	0.63	33.19	146.23
Median i	0.63	1.65	8.19
Quartile 25 i	0.62	1.67	8.21
Min i	0.62	0.35	6.89
Quartile 75 i	0.62	1.78	8.32
Spectral density f	0.59	49.42	308.19
Mean i	0.58	0.61	5.38
Spectral density i	0.57	10.35	356.60
Mean f	0.57	7.03	14.87
Std f	0.51	1.54	9.39
Variance i	0.49	0.77	19.74
Histogram i	0.43	0.31	70.04
Spectral entropy f	0.43	515.95	523.79
Zero crossings i	0.39	0.97	5.08
Spectral flux f	0.39	176.67	184.51
Magnitude^2 i	0.36	0.19	78.48
Entropy f	0.36	7.26	15.10
Magnitude f	0.36	3.52	55.89
Spectral centroid f	0.34	174.93	182.77
Spectral magnitude area^2 i	0.34	32.29	38.91
Spectral magnitude area f	0.34	170.32	172.93
Spectral maxima i	0.32	31.87	41.35
Spectral maxima f	0.30	143.51	151.35

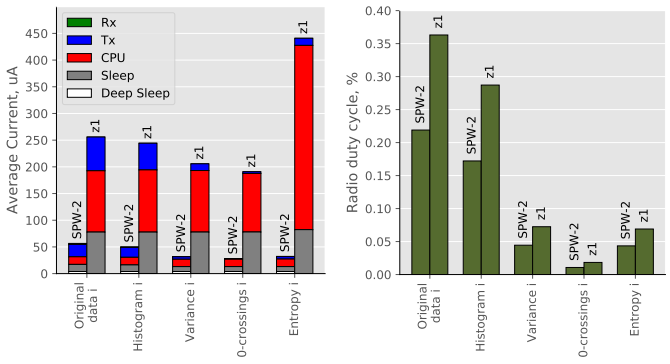
mine the activity. As mentioned previously, RF is used for the classification.

First, we use the fact that RFs provide a means of determining the "importance" of a feature by measuring the total node impurity decrease, weighted by the probability of reaching it, averaged over all of the trees in the Random Forest. The results for this can be seen in Fig. 6. Interestingly, there are similarities between the importance of the best features in the RF have a similar ranking to those determined via measuring their MI.

Second, we show the performance of each individual feature against the associated energy cost. For each feature a randomized search is performed over parameter space to optimize important parameters for the RF. Table 2 demonstrates that some of the best performance is shown by lowest energy cost features, such as the "max" feature. However, the second-best performing features are the spectral histograms, which consume considerably more power. Many other high energy cost features have considerably worse performance on the dataset. Finding a subgroup of features with the best combined cost/accuracy score is a future work item.

5.3 System-level evaluation

Next, we perform a system-level evaluation of on-board processing on SPW-2 and Zolertia Z1. In particular, we set the ADXL362 accelerometer of SPW-2 and the ADXL345 accelerometer of Z1 to generate data at 25 Hz frequency and store the samples in their internal FIFO buffers. The MCU is collecting the samples periodically and extract different time-domain features. The samples generated by the accelerometer are integers: one 8-bit integer per axis. Subsequently, the output data is communicated from the wearable to a receiver using the IEEE 802.15.4-2015 TSCH standard



(a) Average current, μA **(b)** Radio duty cycle, %

Figure 7: Evaluation of the system-level current consumption and radio duty cycle on the SPW-2 platform. On x axis: several representative time domain features arranged by their computational complexity. On y axis: the cumulative energy and duty cycle of an application that samples accelerometer and computes & transmits the single specified feature. This energy consumption is directly proportional to battery lifetime, assuming a linear discharge model and negligible self-discharge.

at 2.4 GHz. At the link layer, the system uses a broadcasting communication scheme that resembles the undirected non-connectable advertisements of Bluetooth Low Energy (BLE), which is a popular option for transporting wearable data. The radio duty cycle depends on the amount of data that need to be communicated after the on-board processing stage: we use an adapted version of the *6tisch* minimal schedule with slotframe size 17 and transmit-only slots.

We compare the system-level energy costs for on-board extraction of four time-domain integer features, namely: histogram, variance, zero-crossing rate and entropy with transmitting the raw data directly (labeled *original data*). Figure 7a plots the long-term average current draw for each scenario, and shows that the long-term average current of the wearable system can be significantly reduced with on-board processing, especially on SPW-2. The savings are primarily due to the reduction of the amount of data the are transmitted, which leads to a significant reduction of the radio duty cycle, as shown in Figure 7b. The cost of using the MCU for additional on-board processing, on the other hand, is negligible on the SPW-2, and is countered by the fact that after the data are reduced in size, fewer MCU operations need to be spent on other operations, for example, for framing the data in packets and moving them from RAM to the output buffer of the radio. The secondary reason for the savings is the very efficient deep-sleep mode on SPW-2, in which it consumes only 4.0 μA [5], on par with the best *msp430* prototypes, and an order of magnitude less than Z1 during its sleep mode with a wake-up timer running [24].

6 Conclusion

In this paper, we demonstrated that on-board extraction of features on modern low-power wearables is both feasible and beneficial for increased system lifetime. In terms of feature comparison, our result show that simple time-domain features have the best cost/benefit properties. We hope the results will stimulate increased research activity in using low-power wearables for more than simple collection of data.

7 Acknowledgements

This work was performed under the SPHERE IRC, funded by the UK EPSRC Grant EP/K031910/1.

8 References

[1] B. Agarwal, A. Chakravorty, T. Wiktorski, and C. Rong. Enrichment of machine learning based activity classification in smart homes using ensemble learning. In *IEEE/ACM UCC*, pages 196–201, 2016.

[2] A. Avci, S. Bosch, M. Marin-Perianu, R. Marin-Perianu, et al. Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. In *ARCS*, pages 1–10, 2010.

[3] C. Bormann and P. Hoffman. Concise Binary Object Representation (CBOR). RFC 7049, IETF, 2013.

[4] D. Chu, N. D. Lane, T. T.-T. Lai, C. Pang, X. Meng, Q. Guo, F. Li, and F. Zhao. Balancing energy, latency and accuracy for mobile sensor data classification. In *ACM SenSys*, pages 54–67, 2011.

[5] X. Fafoutis, A. Vafeas, B. Janko, R. S. Sherratt, et al. Designing wearable sensing platforms for healthcare in a residential environment. *EAI Endorsed Trans. on Pervasive Health and Technology*, 17(12), 9 2017.

[6] A. Hermanis, R. Cacurs, et al. Wearable sensor system for human biomechanics monitoring. In *EWSN*, pages 247–248, 2016.

[7] J. D. Hester, T. Peters, et al. Amulet: An energy-efficient, multi-application wearable platform. In *ACM SenSys*, pages 216–229, 2016.

[8] H. Huang and S. Lin. Toothbrushing monitoring using wrist watch. In *ACM Sensys*, pages 202–215, 2016.

[9] K. Iswandy and A. Koenig. Feature selection with acquisition cost for optimizing sensor system design. *Advances in Radio Science*, 4:135–141, 2006.

[10] U. Jensen, P. Kugler, M. Ring, and B. M. Eskofier. Approaching the accuracy–cost conflict in embedded classification system design. *Pattern Analysis and Applications*, 19(3):839–855, 2016.

[11] J. Ko, K. Klues, C. Richter, W. Hofer, B. Kusy, M. Bruenig, et al. Low power or high performance? a tradeoff whose time has come (and nearly gone). In *EWSN*, pages 98–114. Springer, 2012.

[12] J. Ko, C. Lu, M. B. Srivastava, J. A. Stankovic, A. Terzis, and M. Welsh. Wireless sensor networks for healthcare. *Proceedings of the IEEE*, 98(11):1947–1960, 2010.

[13] A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating mutual information. *Physical Review E*, 69:066138, Jun 2004.

[14] X. Liu, L. Liu, S. J. Simske, and J. Liu. Human daily activity recognition for healthcare using wearable and visual sensing data. In *IEEE ICHI*, pages 24–31, 2016.

[15] A. U. Nambi SN, L. Gonzalez, and V. Prasad R. Coachme: Activity recognition using wearable devices for human augmentation. In *EWSN*, pages 174–179, 2017.

[16] Nordic Semiconductor. nRF52832 Product Specification v1.1, 2017.

[17] Raspberry Pi (Trading) Ltd. Compute Module Datasheet, 2016.

[18] A. Stisen, H. Blunck, S. Bhattacharya, et al. Smart devices are different: Assessing and mitigating mobile sensing heterogeneities for activity recognition. In *ACM Sensys*, pages 127–140, 2015.

[19] M. Tan. Cost-sensitive learning of classification knowledge and its applications in robotics. *Mach. Learn.*, 13(1):7–33, 1993.

[20] N. Twomey, T. Diethe, M. Kull, H. Song, M. Camplani, S. Hannuna, X. Fafoutis, et al. The SPHERE challenge: Activity recognition with multimodal sensor data. *arXiv preprint arXiv:1603.00797*, 2016.

[21] Y. Vaizman, K. Ellis, and G. R. G. Lanckriet. Recognizing detailed human context in-the-wild from smartphones and smartwatches. *IEEE Pervasive Computing*, 2017.

[22] Y. Weiss, Y. Elovici, and L. Rokach. The cash algorithm-cost-sensitive attribute selection using histograms. *Elsevier Information Sciences*, 222:247–268, 2013.

[23] S. Zhang, A. Rowlands, P. Murray, and T. Hurst. Physical activity classification using the GENEA wrist-worn accelerometer. *Medicine & Science in Sports & Exercise*, 44(4):742–748, 2012.

[24] Zolertia. Z1 Datasheet, 2010.

[25] Zolertia. Zolertia Zoul Revision A Internet of Things hardware wireless module, for 2.4-GHz and 863-950MHz IEEE 802.15.4, 6LoWPAN and ZigBee Applications, 2017.